Acta Crystallographica Section D
Biological Crystallography
ISSN 0907-4449

Statistical pattern recognition for macromolecular crystallographers

Richard J. Morris

European Bioinformatics Institute, Wellcome Trust Genome Campus, Hinxton, Cambridge CB10 1SD, England

Correspondence e-mail: rjmorris@ebi.ac.uk

Received 17 December 2003
Accepted 19 August 2004

A selection of pattern-recognition techniques is presented with a special focus on those methods that may be of interest to macromolecular crystallographers not indifferent to automated protein model building. An overview of the most common pattern-recognition approaches is given and some popular model-building packages are briefly reviewed within this context.

1. Introduction

Pattern recognition is perhaps the most important process in the acquisition of knowledge and the driving force behind the development of any scientific discipline. Recognizing behavioural, morphological, genetic *etc*. similarities within and between species (sociology and psychology, biometrics, biology *etc.*), capturing the patterns observed in experimental data in the form of rules and equations (chemistry, physics *etc.*) and many areas of art and tasks of everyday life may be seen as applied pattern recognition.

Indeed, the human mind seems to be intrigued by patterns in nature and the urge to discover these regularities and especially to classify objects possibly provides the satisfaction that drives the artistic and scientific mind to creative thinking and understanding. Pattern recognition is one of the most highly developed and frequently used cognitive capabilities. The brain is considered to be the most sophisticated and powerful pattern-recognition machine available and will probably remain so.

Although pattern recognition in this broad sense is perhaps the most intriguing and interesting, the definition given here will be more restricted and will concentrate mainly on the computer-science applications of machine learning and automatic machine classification.

This paper deals with the artificial intelligence approach to pattern recognition, how to teach a computer to distinguish relevant signals from noise and how to use this information to make decisions. One important field of artificial intelligence for which many of these approaches were developed is that of computer vision and many textbooks tend to focus on image-analysis applications such as handwriting recognition. The underlying theory is, however, sufficiently general to be applied to wide range of important problems.

Especially with the recent explosion of biological data from various genomics projects, there is an increasing interest in pattern-recognition (data-mining) techniques. However, the focus here will be on the application of such techniques to automated model building in protein crystallography. Many model-building packages now employ fairly sophisticated pattern techniques ranging from template-fitting procedures

© 2004 International Union of Crystallography
Printed in Denmark – all rights reserved

to neural networks and likelihood-based image matching. A brief review of such techniques therefore seems quite timely and is the purpose of this article. For a much more detailed and complete presentation of the underlying theory and other methods, the reader should refer to the many pattern-recognition textbooks and papers cited throughout the text. The classics by Duda *et al.* (2003) and Fukunaga (1990) can be highly recommended, as can the excellent book on machine learning and information theory by MacKay (2003).

2. Statistical pattern recognition

One primary goal of pattern recognition is to reduce a wealth of information to a small set of important characteristics (features) that are relevant to the questions being asked and the interpretation of these features: the transformation of signals to meanings. A general model for pattern recognition is depicted in Fig. 1. In statistical pattern recognition, a pattern is represented by a set of d features, $f_1, f_2, \ldots, f_d$, called a feature vector in d-dimensional feature space. For example, in assigning/recognizing secondary-structure elements from a set of coordinates, one could include the following features: f_1, the main-chain backbone angle φ; f_2, the main-chain backbone angle ψ; f_3, the number of residues to the hydrogen-bonding partner of the main-chain O atom; f_4, the number of residues to the hydrogen-bonding partner of the main-chain N atom. The actual feature values are denoted here as $\mathbf{x} = (x_1, \ldots, x_d)^T$. The determination of this set of features is known as feature extraction and feature selection. Following data processing and feature selection, a decision-making process is needed that will take a given feature vector and assign it to one of the predefined classes $\omega_1, \omega_2, \ldots, \omega_c$. c is the number of such classes, *i.e.* the cardinality of the classification set Ω. These classes may be defined either from previous knowledge and the mapping from the feature vector to each class determined with methods known as supervised learning or from the use of clustering techniques (unsupervised learning, exploratory data analysis). Methods for these tasks will be outlined below. For more detailed information and implementation details, the reader is referred to the literature given throughout the text.

2.1. Feature extraction and feature selection

Feature extraction refers to techniques to create new features from transformations of the originally chosen ones. Feature selection refers to algorithms that attempt to select the best subset of features from a given set with the aim of reducing the feature space size. The terms feature extraction and feature selection are, however, often used interchangeably to describe the process of dimensionality reduction. The performance of a classifier system depends on the training sample size, the number of features and the complexity of the target classification. Computational efficiency is strongly dependent on the dimensionality of the problem. It is also often the case that the required information for a particular problem is a small fraction of the full information content one has at hand. For example, the full diffraction experiment information, containing a huge number of photon counts per detector pixel for every image and over all images as well as all boundary conditions, is reduced to a much smaller number of features, consisting of a unique set of Miller indices together with integrated intensity values and error estimates that summarize the information of a particular experiment. It is generally desirable to describe the system of interest with the smallest number of (preferably independent) features possible that is still sufficient to perform the classification without loss of accuracy. In addition, problems in overfitting often mean that a dimensionality reduction actually achieves a decrease in classification error rates. The process of feature selection essentially reduces data redundancy. This process may be seen as one of data compression and information theory and indeed the underlying mathematics are the same (MacKay, 2003). Information compression can be viewed as a mapping from a space of dimension d onto a lower dimensional space, $\mathbf{y} = \mathcal{M}(\mathbf{x})$, whereby the original features $\mathbf{x} = (x_1, \ldots, x_D)^T$ are mapped onto a set of new features $\mathbf{y} = (y_1, \ldots, y_d)^T$. For visualization purposes of high-dimensional data, a sensible dimensionality reduction is of great importance. The standard techniques of feature extraction/selection restrict themselves to looking for the best linear transformation because of the computational advantages that linear systems offer. In such linear cases, the mapping can be represented by $\mathbf{y} = \mathbf{M}^T\mathbf{x}$, where $\mathbf{M}$ is a $D \times d$ matrix (often with $d << D$). Methods such as neural networks and support-vector machines use so-called kernel functions to create a nonlinear mapping. With the right choice of kernel functions, these methods can achieve a high compression rate and/or a good separation of classes in the new feature space. These methods will be outlined in §3.

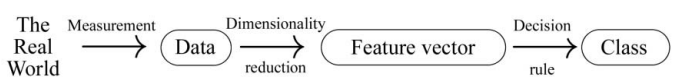

Figure 1
Standard model of a pattern-classification system.

2.2. General mathematical framework for classification

Classification may be described as a decision rule, $\mathcal{C}$, that says for the (commonly real) values $x_1, x_2, \ldots, x_N$ of the features $f_1, f_2, \ldots, f_N$ the pattern belongs to a certain predefined class, ω_i,

$$\mathcal{C} : IR^N \rightarrow \Omega, \qquad \mathcal{C}(\mathbf{x}) = \omega_i.$$

As with most decent statistical theories, any decision making is left to Reverend Thomas Bayes. Bayes' theorem can be written as

$$P(\omega_k|\mathbf{x}) = P(\omega_k)p(\mathbf{x}|\omega_k)/\sum_i P(\omega_i)p(\mathbf{x}|\omega_i),$$

in which, as above, the ω_i parameters are the possible classes and $\mathbf{x}$ is the feature vector. $P(\omega_k|\mathbf{x})$ is the posterior probability, *i.e.* the conditional probability that given the data $\mathbf{x}$ the class is ω_k. $P(\omega_k)$ is the prior belief in this class before observing the features and $p(\mathbf{x}|\omega_k)$ is the likelihood that the class ω_k could produce the observed features. As an example, imagine fitting

side chains to a main chain obtained, for instance, from skeletonization (Greer, 1974). Without looking at the density, the prior belief of a certain C^{α} position having an histidine side chain attached to it could be assigned the frequency of histidine within the protein sequence. The likelihood function could be computed from the fit between the observed and the theoretical electron-density distribution.

The actual decision rule is often formulated as minimizing the risk, $R(\omega_k|\mathbf{x})$, of a false class assignment,

$$R(\omega_k|\mathbf{x}) = \sum_{i=1}^{c} L(\omega_k, \omega_i) P(\omega_i|\mathbf{x}),$$

in which $L(\omega_k, \omega_i)$ is the loss in deciding for ω_k when the true class would be ω_i and $P(\omega_i|\mathbf{x})$ is the posterior probability as calculated above. A loss function is especially important for medical applications, in which a false diagnosis may be fatal and must be excluded at all costs, or for stock-market predictions that may lead to financial disaster. In the simple case of a binary loss function, $L(\omega_k, \omega_i) = \delta_{ij}$, the minimum-risk decision rule reduces to the maximum posterior rule (maximum *a posteriori*, MAP), which states that $\mathbf{x}$ should be assigned to class ω_k if

$$P(\omega_k|\mathbf{x}) \geq P(\omega_i|\mathbf{x}) \quad \forall i \neq k.$$

Basically, all aspects of structure determination including crystal detection and alignment, diffraction-spot recognition, intensity-profile determination, crystal space-group recognition, unphased and phased molecular and micromolecular replacement, heavy-atom detection, model building, refinement and validation can be unified within this framework. Indeed, statistical pattern recognition may be seen as just another instance of applied Bayesian statistics, the use of which has been advocated for years by Jaynes (see references in Jaynes, 2003) and in crystallography by Bricogne and others (French & Wilson, 1978; French & Oatley, 1978; Bricogne, 1988, 1997*a*,*b*).

2.3. A brief look at other approaches to pattern recognition

A distinction is often made between statistical pattern recognition and techniques such as template matching, the so-called syntactic approach (Fu, 1982), neural networks (Bishop, 1995) and support-vector machines (Vapnik, 1998). This distinction is, however, somewhat artificial (Fu, 1983; Schurmann, 1996). Template matching is one of the simplest and earliest methods of pattern recognition. It requires the availability of some known template (typically a shape) and a similarity measure. Often, the template is itself learned from a training set. In its standard form (non-deformable templates, unflexible similarity measure), template matching has the disadvantage of being sensitive to distortions and is computationally relatively demanding. The syntactic approach adopts a hierarchical perspective in which a pattern is viewed as consisting of an arrangement of simpler sub-patterns. Complex patterns are thus built up out of a small number of such sub-patterns and grammatical rules which describe how such sub-patterns may be assembled. A common problem is the identification of suitable sub-patterns, especially in noisy data, and the often combinatorial explosion of ways in which these sub-patterns can be combined. Artificial neural networks have, despite many controversies regarding their usefulness and range of applicability, maintained a high research interest as models for trying to understand the biology of the brain and as powerful classification tools. Support-vector machines are a fairly recent development that in many areas offer stiff competition to neural networks.

3. Overview of some standard techniques

3.1. Parzen windows

To carry out any classification within the Bayesian framework, the prior probabilities and the likelihoods must be known. The Parzen window method (Fukunaga, 1990) estimates the class-conditional probability density function $p(\mathbf{x}|\omega_i)$ (likelihood) by centring kernel functions at each data point $\mathbf{x}^n$ of class ω_i and summing them. No assumption is made about the general functional form of the likelihood (the use of kernel functions is a local spread estimate and is in principle just a histogram binning method); the method is therefore termed non-parametric. Parametric methods, on the other hand, make some strong assumption about the likelihood function. For instance, the structure-factor distribution in the complex plane is often assumed to be Gaussian.

In general, kernel functions will have some sort of metric that measures the distance from each kernel centre (the data point) and a spread attached to it that reflects the sparseness of the data points (the spread should decrease with increasing number of data points).

The most common kernel functions are hypercubic, hyperspheric, Gaussian and exponential. Common metrics are the Euclidean $d(\mathbf{x}, \mathbf{x}^n) = |\mathbf{x}^n - x_j|$, quadratic $d(\mathbf{x}, \mathbf{x}^n) = (\mathbf{x}^n - \mathbf{x})^T\mathbf{M}(\mathbf{x}^n - \mathbf{x})$ and the Chebycheff metric $d(\mathbf{x}, \mathbf{x}^n) = \max|x_j^n - x_j|$.

The performance of the Parzen window method for estimating the likelihood function depends strongly on the distance metric, the window function (shape) and the window size (Fig. 2).

3.2. Principal component analysis

Principal component analysis (PCA), also known as the Karhunen–Loéve expansion, is one of the best known unsupervised linear feature-extraction methods. It is a simple yet powerful technique and other important methods such as factor analysis, correspondence analysis, discriminant analysis or kernel PCA may be viewed as variations on the same theme. PCA will therefore be presented in greater detail than the other techniques.

PCA seeks the best summary of the data (see Fig. 3), followed by the second best and so on. This process is carried out by looking for new axes that can explain the maximal amount of variance in the data (the axes that fit the data best in the least-squares sense). For example, if one could find a straight line that passes through all the points in a plot, then this new coordinate axis would be sufficient to describe the

data exactly and all axes would be redundant, *i.e.* the total variance of the data could be accounted for (summarized) by the first principal component. Fig. 1 shows a two-dimensional plot of data and a new axis (the first principal component) that fits through these points best in terms of squared deviation. Introducing a second axis in Fig. 1, orthogonal to the first principal component, would allow one to account for the total variance in the original data and would therefore be information conserving. The distribution of information between the axes has, however, been greatly shifted compared with the original coordinate system (the features x_1 and x_2) and most of the information now resides in only one projection (the principal component axis y_1). Without too much information loss, y_1 may be sufficient to describe the data for classification purposes. This approach can often lead to a substantial dimensionality reduction in real applications. It is common to introduce PCA as a recursive procedure in which one looks for the direction of first principal component, $\mathbf{e}_1$, such that this vector passes through the sample mean and minimizes the squared error from projecting the data points onto this axis. The first principal component thus gives the direction (vector) in feature space that contains the most information about the data. Further principal components are sought in a similar manner of minimizing the squared error of the projections of data points onto the new axis. This recursive feature-extraction procedure is terminated once an information-loss threshold is met. Keeping the original dimensionality of feature space, $d = D$, PCA is an information-conserving transform. If one reduces the feature space of original dimension D to d, then the reconstruction error is given by

$$\varepsilon = \sum_{i=d+1}^{D} \lambda_i.$$

This can expressed as the loss of information in percentage by

$$\varepsilon = \left(\sum_{i=d+1}^{D} \lambda_i \Big/ \sum_{i=1}^{d} \lambda_i \right) \times 100\%.$$

PCA, however, need not be performed in the above form. The recursive procedure can be shown to be equivalent to an eigenanalysis on the covariance matrix,

$$\mathbf{C} = \begin{bmatrix} \mathrm{Cov}(x_1, x_1) & \mathrm{Cov}(x_1, x_2) & \dots & \mathrm{Cov}(x_1, x_n) \\ \mathrm{Cov}(x_2, x_1) & \mathrm{Cov}(x_2, x_2) & \dots & \mathrm{Cov}(x_2, x_n) \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{Cov}(x_n, x_1) & \mathrm{Cov}(x_n, x_2) & \dots & \mathrm{Cov}(x_n, x_n) \end{bmatrix},$$

with

$$\mathrm{Cov}(x_i, x_j) = \mathcal{E}\{(x_i - \mu_i)(x_j - \mu_j)\} = \mathcal{E}\{x_i x_j\} - \mathcal{E}\{x_i\}\mathcal{E}\{x_j\}.$$

The covariance may be estimated from

$$\mathrm{Cov}(x_i, x_j) \simeq \mathrm{cov}(x_i, x_j) = \frac{1}{N-1} \sum_{s=1}^{N} (x_{i_s} - \overline{x_i})(x_{j_s} - \overline{x_j}),$$

where $\overline{x_i}$ denotes the average value over all the data points, x_{i_s}, for the variable x_i, *i.e.* the sample mean.

The eigenvector with the largest eigenvalue is the direction with the largest variance of the projected data and is therefore equal to the first principal component.

A $D \times D$ matrix $\mathbf{C}$ is said to have an eigenvector $\mathbf{e}$ to the eigenvalue λ if $\mathbf{Ce} = \lambda\mathbf{e}$. If one places all these eigenvectors $\mathbf{e}_i$

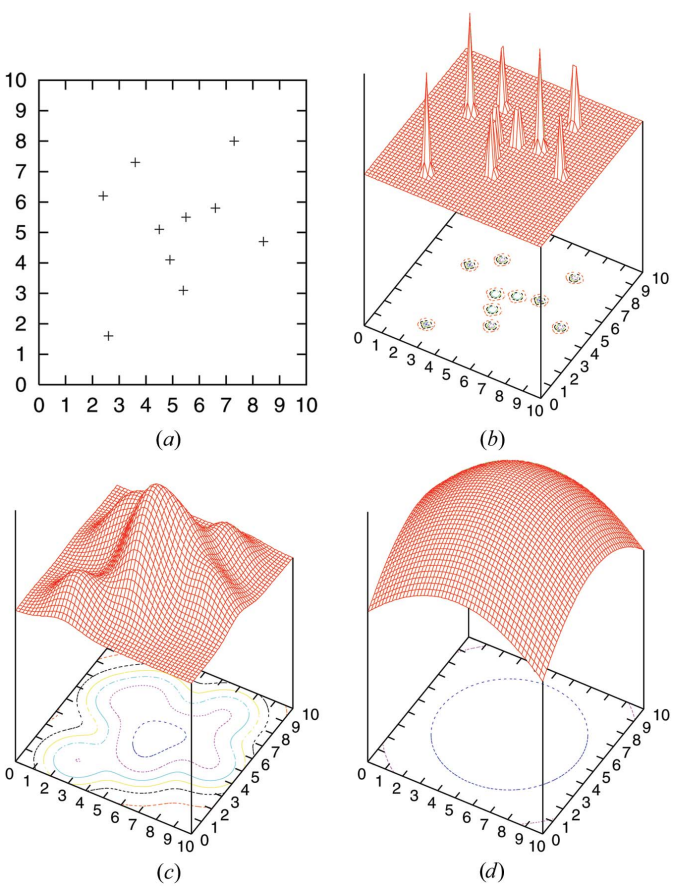

Figure 2
Parzen windows: from data points to a distribution. (*a*) shows ten measured data points. The other three plots show estimations of the distribution obtained by attaching a known function to each data point, *i.e.* a kernel function. The plots show the influence of the choice of window size. Gaussians with $\sigma = 0.1$, 1.0 and 10.0 are depicted here in (*b*), (*c*) and (*d*), respectively.

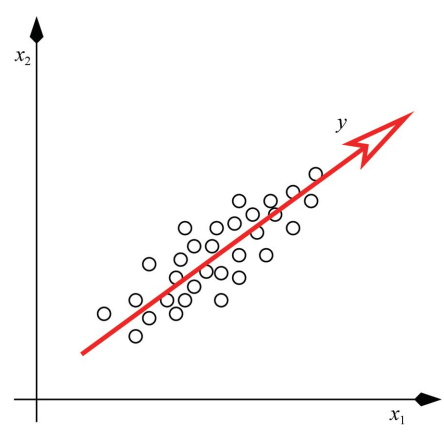

Figure 3
Principal component analysis of a two-dimensional (x_1, x_2) data set. The line labelled y shows the direction of the first principal component, which gives the best mean-square reduction of the data from two to one dimensions.

as columns into one matrix, $\mathbf{M}$, and places all the eigenvalues, λ_i, as the elements of a diagonal matrix, $\widehat{\mathbf{C}}$, then the eigensystem may be written as $\mathbf{CM} = \mathbf{M}\widehat{\mathbf{C}}$. Multiplication of this equation from the left by $\mathbf{M}^{-1}$ leads to

$$\widehat{\mathbf{C}} = \mathbf{M}^{-1}\mathbf{CM}.$$

The transformation matrix that diagonalizes the original covariance matrix contains the new basis vectors (the eigenvectors of the original matrix) as its columns. Thus, an eigenanalysis on $\mathbf{C}$ is equivalent to diagonalization and corresponds to finding new direction vectors such that the new features are uncorrelated. Ordering these vectors by their eigenvalues (equal to the variances) gives the directions of the principal components as outlined in the above recursive procedure. For Gaussian data, the lack of correlation between the new features also guarantees their independence, providing a computationally advantageous way for calculating the joint probabilities, $P(y1, \ldots y_d) = P(y_1)\cdots P(y_d)$, that would otherwise be an often bad approximation.

The main steps of PCA can be summarized as follows.

(i) Choose any number of features, $(f_1, \ldots, f_D)^T$, for the classification problem.

(ii) Construct the covariance matrix, $\mathbf{C}$, of these features from a test set.

(iii) Perform an eigenanalysis on $\mathbf{C}$.

(iv) Order eigenvalues, λ_j, and eigenvectors, $\mathbf{e}_j$. Choose the d first vectors (typically $<< D$) to define the new feature space. These new vectors may be obtained from the original vectors *via* the transformation $\mathbf{y} = \mathbf{M}^T\mathbf{x}$.

If features that are used vary wildly in magnitude it is common practice to replace the covariance matrix by the correlation matrix in all the above arguments.

3.3. Independent component analysis

Although PCA offers a powerful method by which to summarize data, for clearly non-Gaussian distributions this approach is not well suited (see Fig. 4). PCA is a second-order method, which means that only information contained in the covariance matrix of the data is used. If the data are normally distributed then they are completely determined by this second-order information and including higher moments is therefore pointless.

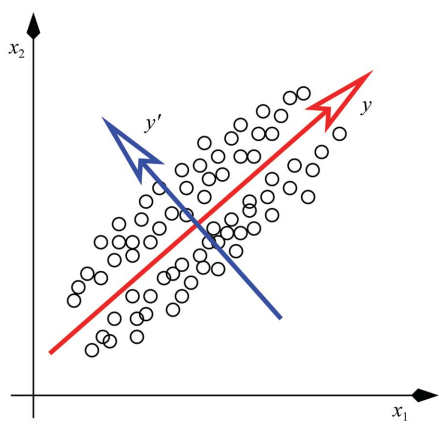

Figure 4
An illustration of the problems with principal components analysis and other variance-based methods. The data could be cleanly separated into two clusters by the direction labelled y', as obtained from linear discriminant analysis (LDA). However, projections along the principal component direction (y) would not produce any such clustering.

As the name implies, independent component analysis (ICA) attempts to find a transformation that produces components that are as statistically independent as possible. Any ICA method thus may roughly be expressed as the definition of an independence measure (objective function) and an algorithm for maximizing it (optimization procedure). A natural quantity to measure the dependence of two variables, X_1 and X_2, is the mutual information (divergence). For discrete variables the divergence is

$$I(X_1, X_2) = \sum_{x_1 \in X_1} \sum_{x_2 \in X_2} P(x_1, x_2) \log P(x_1, x_2)/[P(x_1)P(x_2)]$$

and for continuous variables

$$I(X_1, X_2) = \int_{x_1} \int_{x_2} p(x_1, x_2) \log p(x_1, x_2)/[p(x_1)p(x_2)].$$

The divergence is, however, not a proper distance measure (metric) as the triangle inequality property is not obeyed. Other measures of independence exist, many related to or approximations of the above divergence or the negentropy. ICA is a very general concept with a broad range of applications in many different areas and the method is still very much under development. Linear ICA is perhaps the most developed and several freely available software packages exist that should be given preference over PCA for (highly) non-Gaussian data.

3.4. Discriminant analysis

The Bayesian approach to classification requires knowledge of the class-conditional (likelihood) functions $p(\mathbf{x}|\omega_i)$. Instead of making assumptions about the form of $p(\mathbf{x}|\omega_i)$, one can make assumptions directly about the form of the class boundaries. The decision boundaries can be described by so-called discriminant functions, $g(\mathbf{x})$. For example, in the two-class case a discriminant function may assign $\mathbf{x}$ to class ω_1 if $g(\mathbf{x}) < k$ and to class ω_2 if $g(\mathbf{x}) > k$. On the boundary $g(\mathbf{x}) = k$ the choice is arbitrary. A connection to Bayes' decision theory can be made by setting

$$g(\mathbf{x}) = p(\mathbf{x}|\omega_1)/p(\mathbf{x}|\omega_2)$$

and k to $P(\omega_2)/\mathrm{P}(\omega_1)$. This is indeed an optimal discriminant function, as would be expected from anything Bayesian, but this optimal function is not unique as any monotonic function, f, will lead to exactly the same decision boundaries for $f[g(\mathbf{x})]$ and $f(k)$. In general, one can achieve optimal discriminant functions (equivalent to Bayes' decision rule) by setting $g_i(\mathbf{x}) = p(\mathbf{x}|\omega_i)P(\omega_i)$; it is, however, common to assume a much simpler type of decision function, such as a linear combination of the components of the feature vector $\mathbf{x}$. Linear discriminant functions have the general form

$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0,$$

where $\mathbf{w}$ is called the weight vector and w_0 is the threshold weight (or bias). This equation describes a hyperplane normal to the direction of the weight vector and at a distance $w_0/|\mathbf{w}|$ from the origin. Expressing $\mathbf{x}$ as $\mathbf{x}_\mathrm{p} + d\mathbf{w}/|\mathbf{w}|$, where $\mathbf{x}_\mathrm{p}$ is the normal projection of $\mathbf{x}$ onto the hyperplane defined by g and d is the distance from the plane (negative if on the negative side of the plane, otherwise positive). As $g(\mathbf{x}_\mathrm{p}) = 0$, $g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0 = d|\mathbf{w}|$. The discriminant function for $\mathbf{x}$ thus measures the distance from the hyperplane.

A weight vector that correctly separates classes is often termed a solution vector. Basically, one would like the hyperplane boundaries to be roughly halfway between two classes. One common possibility for determining this vector is to maximize the minimum distance of the sample points to the plane. Another approach is to maximize the margin between the hyperplane and the nearest data points. Other methods include so-called error-correcting procedures that only change one of the current hyperplanes if it caused a misclassification. Examples are the perceptron and relaxation procedures. Error-correcting methods suffer from convergence problems if the samples are not linearly separable; distance and minimum-square error procedures converge, but the resulting weight vector is not necessarily a solution vector.

A pattern-classification method that employs linear discriminant functions is termed a linear machine. The minimum-distance classifier (nearest neighbour rule) is a linear machine. Problems that can successfully be divided up by hyperplanes into different classes are called linearly separable.

By using functions of $\mathbf{x}$, $\varphi_i(\mathbf{x})$, one can use the above idea but with a discriminant function that is nonlinear in the original measurements x (but linear in the new space defined by the functions φ_i). This approach is known as generalized linear discriminant analysis.

3.5. Artificial neural networks

Artificial neural networks are computational models based on simplified models of the biological neuron. From this point of view, a neural network is a mathematical model for the brain. From a computational viewpoint, a neural network is a general machine that represents functions by a network of simple computing elements and a general learning framework.

The nerve cell, or neuron, is a fundamental functional unit of the brain. Basically, each neuron consists of inputs (dendrites), outputs (axons) and a processing unit (soma). The connection between an axon and a dendrite from different cells is called the synapse. A signal is transmitted by releasing chemical substances from the synapses, which interact with receptors on the dendrite, causing a change in the electrical potential of the cell; if a potential threshold (activation) is reached, a pulse spreads down the axon that eventually may pass into other cell bodies, again *via* synapses and dendrites. In the human brain each neuron has been estimated to be connected to up about 10 000 other neurons.

An artificial neural network consists of a number of nodes that are connected by links. Each link has a weight associated with it and it is these weights that provide the learning ability and memory effect. The output links are only allowed to transmit signals if an activation level is reached from combining all incoming signals. Common activation functions include the sign function, linear functions and, perhaps the most frequently used, the sigmoid function. The type of activation function is often indicated within the nodes (Fig. 5).

There are different network architectures, each with different computational properties. A network structure typically has several layers (multilayer network); single-layer networks are called perceptrons. In a feed-forward network the links are always only unidirectional (forward) and only between different layers, *i.e.* no cycles, whereas in a recurrent network there are no such restrictions.

One popular method of training a multilayer network is back-propagation. Examples are presented to the network and the network attempts to classify each one. If the classification is correct then nothing is done, but if the output is erroneous then the weights are adjusted back to share the blame. This can be shown to be a gradient-descent method in weight space (see Jain *et al.*, 2000, and references therein).

For pattern recognition, neural networks often perform very well, but because of their black-box nature it is difficult to extract much more information than the output signal for a given input. Given enough parameters, neural networks have no problem overfitting data (basically memorizing all presented patterns and not making the necessary generalizations). It is, therefore, highly important to use some kind of cross-validation technique. A similar method that can better handle prior knowledge and offer advantages in modelling probabilistic functions is the Bayesian belief networks technique (Russell & Norvig, 1996; MacKay, 2003).

3.6. Kernel methods (support-vector machines)

The search for decision boundaries between different classes is often complicated and hindered by the fact that the data are not well separated and appear jumbled together. However, it is possible to (nonlinearly) map this data onto a higher dimensional space in which they are cleanly separated, at least in principle. The discovery of such a mapping is not a trivial task and often the new dimension can be so high that overfitting can hardly be avoided. However, the power and appeal of methods for the determination of (linear) decision boundaries is great enough to first attempt to find such a

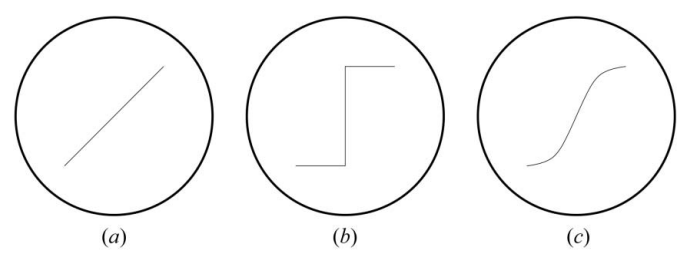

Figure 5
Some typical artificial neural network activation functions. (*a*) Linear, (*b*) sign, (*c*) sigmoid.

mapping and then to use these well established techniques to separate the different classes.

The idea of support-vector machines (SVM) is to maximize the margin, b, between different classes. The classes are separated by hyperplanes. Each hyperplane may be described by its normal vector; the nearest transformed sample points are called support vectors (see Fig. 7). Note that this process is basically equivalent to generalized linear-discriminant analysis, the difference being that an additional mapping to a higher dimensional space takes place before the decision boundaries are determined by standard linear-discriminant analysis. Support-vector machines are still very much under development, but already a number of implementations are emerging, especially for two-class systems, that are capable of outperforming other methods.

4. Automated model building in protein crystallography

Electron-density map interpretation consists of examining the electron-density map and building a molecular model that is based on the features of that map and knowledge of the chemical nature of the molecule. Before the emergence of automated software approaches, this was a labour-intensive task that could typically take many months and required a sound knowledge of protein structure and chemical bonding, experience, imagination and much expertise (Kleywegt & Jones, 1996).

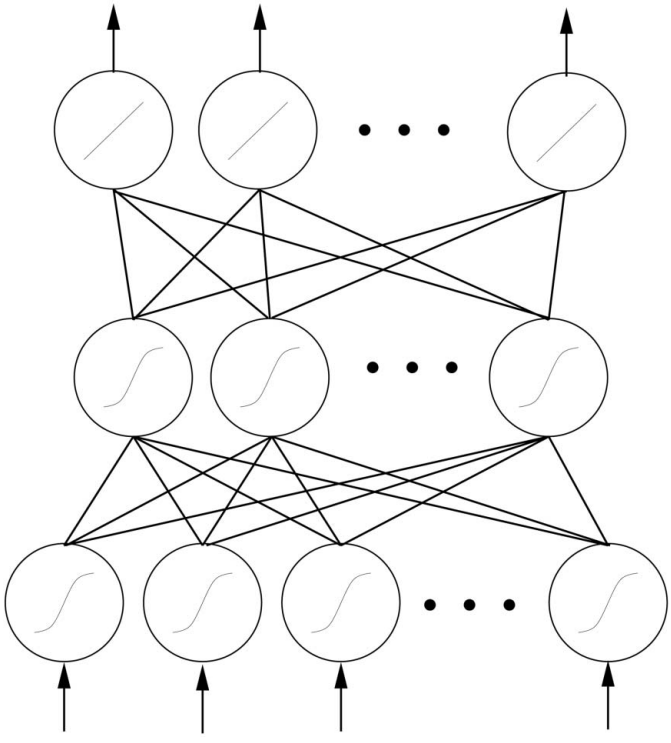

Figure 6
An artificial example of an artificial neural network. Input signals are presented to the system (lower series of arrows); these are combined within the intermediate layer(s) and are again combined in the final layer to create the output (the upper arrows). The output signals can directly map onto the various classes or can be used to create new features that enjoy a better (linear) separability than the original ones.

It is beyond the scope of this article to review all existing model-building ideas and programs and to do many of the excellent approaches justice; therefore, a small number of selected methods will be examined in sufficient detail to get an impression of the pattern-recognition aspect. The current programs have much in common but also subtle differences, either in their philosophy or in their implementation details. Many of the methods below are derivatives of algorithms originally employed in molecular-graphics packages such as *O* (Jones *et al.*, 1991; Jones & Kjeldgaard, 1996), *QUANTA* (Accerlrys Inc.) or *XtalView* (McRee, 1992) to aid the manual model-building process. The selection is naturally biased towards the author's own experience and knowledge of the individual programs and is perhaps not a representative selection of original and historically important ideas. The ordering is meant to reflect neither historical developments nor any quality assessment. For a detailed comparison of some of the most popular automated model-building packages see Badger (2003). Further details should be sought in the original articles and in the *Proceedings of the CCP4 Study Weekend* (2004).

4.1. Greer's skeleton and Jones' bones

Greer (1974) proposed a very simple and elegant method to reduce the complexity of a three-dimensional electron-density distribution to a much clearer set of lines that capture the connectivity (main chain) of the map. The method is an iterative procedure that removes points from a grid representation of the electron density. In short, points are only removed if they do not break the connectivity. One is left with a small set of points that can be joined to produce a skeleton of the original density. Although there have been many developments that are more sophisticated, skeletonization remains a powerful and commonly used method and it is perhaps still the most widely employed technique for macromolecular model building in experimental electron density thanks to implementations such as that of the popular graphics package *O* (Jones *et al.*, 1991; Jones & Kjeldgaard, 1996). Similar approaches that in addition make use of the electron-density topology (maxima, minima, saddlepoints) are the core-tracing algorithm of Swanson (1994) and the molecular-scene analysis of Fortier *et al.* (1997).

4.2. The *ESSENS* of Kleywegt, Cowtan's *FFFEAR* and Diller's *DADI*

The *ESSENS* routine (Kleywegt & Jones, 1997) aims to interpret a given electron-density map by attempting to recognize given templates at each point in the map. An exhaustive search in six-dimensional space was proposed, equivalent to phased molecular replacement. This was possibly the first successful approach to automate the idea of building new proteins from old fragments (Jones & Thirup, 1986). At each point in space, one tries to find the best match (classify) to one of a given number of search fragments (classes). The mapping function is based on the worst agree-

ment of the electron density at the atomic centres between the search model and the map density.

One disadvantage of the *ESSENS* implementation is the time required to carry out the six-dimensional search in real space. Cowtan has attacked this issue by re-formulating the translation search in reciprocal space and implemented the approach with FFT methods within the program *FFFEAR* (Cowtan, 1998). In the pattern-recognition context this is an improved classification function.

The idea of *DADI* (Diller, Redinbo *et al.*, 1999) is similar. These authors created a database of frequently observed *CATH* (Orengo *et al.*, 1997) domains and employed a more top-down approach of first recognizing and placing the domains (Diller, Pohl *et al.*, 1999). A novel Monte Carlo chop-and-clip procedure then prunes the built molecule down through the secondary-structure level to individual residues, always only keeping the pieces that fit well. The domain-recognition score is calculated from each atom and is based on the observed electron density, the molecule's internal energy, the electrostatic potential, the van der Waals energy and any protein–ligand interaction energy. The approach is basically equivalent to the idea of *ESSENS* but with a different classification function and automatic methods to choose the search fragments (classes) that are later subdivided to increase the placement accuracy.

4.3. *RESOLVE*

The underlying idea of *RESOLVE* (Terwilliger, 2001, 2003*a*,*b*,*c*,*d*) is basically the same as the template-matching method proposed by Kleywegt & Jones (1997); however, a number of further developments and refinements have greatly improved its performance and popularity. *RESOLVE* is perhaps one of the most advanced programs in terms of implementing a proper statistical pattern-recognition system. Not all routines are fully Bayesian yet, but the employed maximum-likelihood method is certainly a good approximation (and even exact for non-informative priors or the case in which only one model is considered). Terwilliger uses the same pattern-recognition techniques to build the macromolecular model and to carry out density modification. This approach often significantly improves the phase estimates and therefore the electron-density map (Terwilliger, 1999, 2000, 2003*d*). FFT-based methods are employed to compute derivatives of a log-likelihood map. Structure-factor statistics are combined with prior knowledge of density distributions to provide log-likelihood scores that allow likely templates to be assigned to grid points and their surrounding region. This method is similar to the *ARP*/*wARP* approach of overlapping fragments to build the main chain (Morris *et al.*, 2002) but without the intermediate step of atom location and peptide-unit identification. Performed in an iterative procedure (Perrakis *et al.*, 1999; Terwilliger, 2003*a*), *RESOLVE* is probably the most robust method below 2.5 Å. The side-chain placement algorithm is perhaps the first routine to employ a truly Bayesian approach to this problem (Terwilliger, 2003*b*). Another very interesting further development of *RESOLVE* involves the use of template matching of a number of commonly observed density distributions to perform density modification in a map such that the suggested density value is independent of the point in question (Terwilliger, 2003*d*).

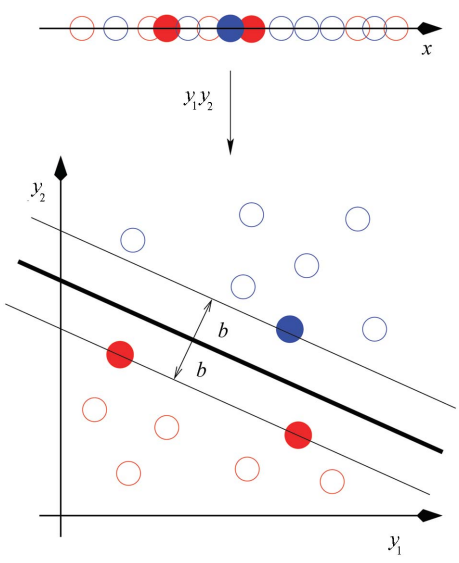

Figure 7
Principal of support-vector machines. Sample points that are not easily separable in the original feature space, x, are transformed to a higher-dimensional space, y_1 and y_2, and then hyperplanes are constructed to maximize the margin between different classes. Depicted is a two-class problem (blue and red). The full points are the support vectors.

4.4. *ARP*/*wARP*

ARP/*wARP* (Perrakis *et al.*, 1999; Lamzin *et al.*, 2001; Morris, Perrakis *et al.*, 2004; Morris, Zwart *et al.*, 2004) uses the syntactic approach of pattern recognition to look for a sub-pattern first. The basic elements that *ARP* (Lamzin & Wilson, 1993) attempts to identify are atoms. These basic elements are used in the template-matching routine (Lamzin & Wilson, 1997) to identify possible peptide planes. A statistical pattern-recognition module then attempts to determine the best path through three-dimensional space by connecting these peptide units such that the resulting main chain matches current geometrical expectations of protein structures (Morris, 2000; Morris *et al.*, 2002). This method is not dissimilar from that proposed by Koch (1974) for the automatic building of simple organic and inorganic molecules, although for proteins extra layers of sophistication are required. The geometrical protein descriptors employed by *ARP*/*wARP* were determined from a principal component analysis of a large set of possible distances and angles (Oldfield & Hubbard, 1994; Kleywegt, 1997) to give a smaller number of better features (Morris, 2000). The distributions were then built using the Parzen window method. The identification of helical substructures is established using linear-discriminant analysis of C^{α}-angle features. The next step in *ARP*/*wARP* is to dock the sequence and fit the side chains. The sequence docking is a pattern-recognition routine based on graph matching that classifies each C^{α} position as belonging to a certain side-chain type

(Morris, 2000; Cohen *et al.*, 2004; Terwilliger, 2003*b*) and may be seen as an adaptation of the method of Zou & Jones (1996). The prior distribution is derived from the given sequence and the likelihood score (class-conditional probability) is an empirical score based on how well the found atoms can reproduce the known connectivity graph for each side chain.

4.5. *TEXTAL* and *CAPRA*

TEXTAL (Holton *et al.*, 2000) operates in a 19-dimensional feature space consisting of electron-density-derived rotationally invariant features. These include the average density and higher moments, the variance over grid points, the moments of inertia and the spokes of density values (the three bonding tubes of density that are to be expected around a C^{α} atom). *TEXTAL* computes these features around a C^{α} position in an unknown structure and attempts to recognize the closest density pattern from a large database of precomputed patterns using diffraction data from 10 to 2.8 Å in *P*1 on an orthogonal grid of 1 Å spacing. Once the density pattern has been matched the coordinates from the database template are used to place atoms in the unknown density. The method relies on the knowledge of the C^{α} positions, a rather severe limitation when starting only from density, but a C^{α} pattern-recognition algorithm (*CAPRA*) has recently been published. *CAPRA* (Ioerger & Sacchettini, 2002) attempts to predict the positions of C^{α} atoms in a given density map. The method employs a variant of the skeletonization algorithm to localize the search space down to the vicinity of the main chain. The skeletonization is performed on a 0.5 Å grid and these points are the candidate positions around which density features are extracted and fed into a neural network. The features that *CAPRA* uses are the same rotationally invariant ones that *TEXTAL* uses. They are computed for 3 and 4 Å spheres around each candidate position, thus providing the standard feed-forward neural network with one hidden layer of sigmoid thresholds with a total of 38 features. The network was trained with the back-propagation method using density values computed to 2.8 Å.

4.6. Levitt's *MAID* and Oldfield's *QUANTA*

The program *MAID* (Levitt, 2001) implements perhaps what comes the closest to human expertise in model building and the approach is clearly based on the steps followed by an experienced macromolecular crystallographer. *MAID* also relies on skeletonizing a given electron-density map to determine the path of the main chain. The skeleton grid points then serve as features in which secondary-element patterns (φ, ψ) are sought, very much like the computer graphics model-building steps with a program such as *O* (Jones *et al.*, 1991). Routines then build in a stereochemically accurate atomic model in these secondary-structure regions, whilst taking care not to over-interpret. Built secondary-structure main-chain fragments of sufficient length (15–20 residues) are slid along the sequence and the best match to the side-chain density is used to assign the residue type. Loops are built by extending the built main-chain fragments by sampling Ramachandran space.

The algorithms developed by Oldfield (2002*a*, 2003) for model building also rely on the reduction of electron density to form so-called bones. Additional methods for ligand fitting and atomic resolution model building exist that employ pattern recognition, graph theory and torsion-angle refinement (Oldfield, 1996, 2001*a*,*b*, 2002*b*). The method for recognition of possible C^{α} positions is very similar to Greer's original description of the algorithm (Greer, 1974) as branches along the skeleton. However, additional artificial branch points are placed to ensure that a sufficient number of points are sampled and an elaborate set of decision rules are employed to enhance the accuracy of the C^{α} position determination (Oldfield, 2003). The bone points are handled as a tree and efficient graph algorithms are employed for the analysis. A depth-first routine searches the tree and for every unit a principal component analysis is performed on the inertia tensor calculated from all the points in that unit. The eigenvalues are then used to classify the part of the structure as strand or helix. These elements are then used as a starting point for tracing methods based on a depth-first search and a number of heuristics. The above algorithms are an integral part of *QUANTA* (Accelrys Inc.). The efficient implementation allows fast execution and this combined with the interactive graphics capabilities makes *QUANTA* perhaps overall one of the most advanced model-building tools currently available.

4.7. Bricogne's micromolecular-replacement method

All the above approaches may be considered implementations and heuristic approximations of the micromolecular-replacement method (Bricogne, 1994, 1995, 1997*a*,*b*). This approach provides a general framework for the recognition of any chosen unit from heavy-atom detection to full molecular replacement and includes all available knowledge at each stage. If phases are available, this approach can be run in the 'Fourier regime' to provide a general framework for model building with a set of small protein fragments. For this purpose a database of frequently occurring fragments of various lengths has been constructed. These fragments were determined with a novel structure-alignment algorithm (Morris, unpublished work) based on the dynamic programming idea of sequence alignment. This micromolecular-replacement method is still under development but has already shown some initial success in placing small fragments (Blanc, unpublished work). Pavelcik *et al.* (2002) have independently implemented a similar approach for placing fragments and have used it successfully with convincing results in a number of test cases.

5. Summary and outlook

A very brief survey of some important pattern-recognition techniques has been presented. Although the level is too basic to be useful for developers, it is hoped that some commonly occurring techniques have been slightly demystified for those

new to the field and that a pattern-recognition awareness has been created. For all methods presented here, well tested and sometimes even well documented freely available software packages exist (such as '*R*'; Ihaka & Gentleman, 1996) that can be used to experiment with and quickly test ideas. With a few exceptions, the implementation of these methods is not challenging for anyone with programming experience and once the method of choice has been determined it is often of advantage in terms of speed and overall performance to write code that is more specific to the problem at hand.

Some of the currently popular software packages and theoretical approaches to automated protein model building into X-ray diffraction electron-density maps have been mentioned and their underlying pattern-recognition techniques have been highlighted. Although many good ideas and fairly sophisticated pattern-recognition techniques are now being applied to the problem of automated model building, the methods often seem far from optimal and contain many heuristics. The implementation of advanced techniques combined within a Bayesian framework for decision making will undoubtedly lead to a higher degree of automation and to model-building systems that are more robust and reliable. In particular, the feature-extraction and selection stage for identifying atoms, mainly C^{α} atoms, peptide units and larger commonly occurring fragments would benefit from more powerful methods that work at different resolutions and electron-density maps of various quality. Although the iterative combination of model building and refinement can often overcome poor initial phase estimates with sufficiently high-resolution good-quality data, in general low or even missing local density naturally presents a problem. Predictive model building based on previously observed patterns combined with hypothesis testing, again *via* refinement, may help to push the limits in terms of data quality, resolution and phase quality a little further. Of potentially great importance will be the integration of structural database information and especially of validation tools directly into the model-building process. All relevant information should be readily accessible at all stages of structure solution and further analysis.

The proper utilization of the overwhelming and constantly growing amount of information in the biological sciences will itself depend heavily on the development of better pattern-recognition techniques. These may be employed for hypothesis-driven data analysis or more challengingly to scan automatically all available literature and all databases to form new hypotheses and to drive the imagination of future researchers. It is anticipated that such techniques will also slowly creep into various stages of structure determination and especially the data mining of (not only) structural information.

I am grateful for financial support from EU grant No. BIO2-CT920524/BIO4-CT96-0189 during the duration of my PhD at the EMBL Hamburg Outstation, for financial support from SPINE, QLG2-CT-2002-00988, during the preparation of the manuscript, to Gérard Bricogne for the exposure to Bayesian statistics and many educating mathematical discussions, to Anastassis Perrakis for many fruitful and interesting discussions and especially to Victor Lamzin for directing my research towards the problems in automated protein model building, many ideas and advice. I thank Janet Thornton, Tom Oldfield and Rafael Najmanovich for constructive critique on an earlier draft of this manuscript and one anonymous referee for helpful suggestions that have hopefully improved the clarity of presentation.

References

Badger, J. (2003). *Acta Cryst.* D**59**, 823–827.
Bishop, C. M. (1995). *Neural Networks for Pattern Recognition.* Oxford: Clarendon Press.
Bricogne, G. (1988). *Acta Cryst.* A**44**, 517–545.
Bricogne, G. (1994). *ACA Transactions Symposium: Likelihood, Bayesian Inference and their Application to the Solution of New Structures*, Abstract TRN14, p. 41.
Bricogne, G. (1995). *ECCC Computational Chemistry*, edited by F. Bernardi & J. L. Rivail, pp. 449–470. Woodbury, New York: American Institute of Physics.
Bricogne, G. (1997*a*). *Methods Enzymol.* **276**, 361–423.
Bricogne, G. (1997*b*). *Methods Enzymol.* **277** 14–18.
Cohen, S., Morris, R. J., Fernandez, F., Ben Jelloul, M., Kakaris, M., Parthasarathy, V., Kleywegt, G., Lamzin, V. S. & Perrakis, A. (2004). Submitted.
Cowtan, K. (1998). *Acta Cryst.* D**54**, 750–756.
Diller, D. J., Pohl, E., Redinbo, M. R., Hovey, T. & Hol, W. G. J. (1999). *Proteins*, **36**, 512–525.
Diller, D. J., Redinbo, M. R., Pohl, E. & Hol, W. G. J. (1999). *Proteins*, **36**, 526–541.
Duda, R. O., Hart, P. E. & Stork, D. G. (2003). *Pattern Classification*, 2nd ed. New York: Wiley-Interscience.
Fortier, S., Chiverton, A., Glasgow, J. & Leherte, L. (1997). *Methods Enzymol.* **277**, 131–157.
French, S. & Oatley, S. (1978). *Crystallographic Statistics*, edited by S. Ramaseshan, M. F. Richardson & A. J. C. Wilson, pp. 19–51. Bangalore: Indian Academy of Sciences.
French, S. & Wilson, K. S. (1978). *Acta Cryst.* A**34**, 517–525.
Fu, K. S. (1982). *Syntactic Pattern Recognition and Applications.* Englewood Cliffs, NJ, USA: Prentice-Hall.
Fu, K. S. (1983). *IEEE Trans. Pattern Anal. Mach. Intell.* **5**, 200–205.
Fukunaga, K. (1990). *Introduction to Statistical Pattern Recognition*, 2nd ed. New York: Academic Press.
Greer, J. (1974). *J. Mol. Biol.* **82**, 279–301.
Holton, T., Ioerger, T. R., Christopher, J. A. & Sacchettini, J. C. (2000). *Acta Cryst.* D**56**, 722–734.
Ihaka, R. & Gentleman, R. (1996). *J. Comput. Graph. Stat.* **5**, 299–314.
Ioerger, T. & Sacchettini, J. C. (2002). *Acta Cryst.* D**58**, 2043–2054.
Jain, A. K., Duin, R. P. W. & Mao, J. (2000). *IEEE Trans. Pattern Anal. Mach. Intell.* **22**, 4–37.
Jaynes, E. T. (2003). *Probability Theory: The Logic of Science*, edited by L. Bretthorst. Cambridge University Press.
Jones, T. A. & Kjeldgaard, M. (1996). *Methods Enzymol.* **277**, 173–198.
Jones, T. A. & Thirup, S. (1986). *EMBO J.* **5**, 819–822.
Jones, T. A., Zou, J.-Y., Cowan, S. W. & Kjeldgaard, M. (1991). *Acta Cryst.* A**47**, 110–119.
Kleywegt, G. J. (1997). *J. Mol. Biol.* **273**, 371–376.
Kleywegt, G. J. & Jones, T. A. (1996). *Acta Cryst.* D**52**, 829–832.
Kleywegt, G. J. & Jones, T. A. (1997). *Methods Enzymol.* **277**, 208–230.
Koch, M. H. J. (1974). *Acta Cryst.* A**30**, 67–70.

Lamzin, V. S., Perrakis, A. & Wilson, K. S. (2001). *International Tables for Crystallography*, Vol. *F*, edited by M. Rossmann & E. Arnold, pp. 720–722. Dordrecht: Kluwer Academic Publishers.

Lamzin, V. S. & Wilson, K. S. (1993). *Acta Cryst.* D**49**, 129–149.

Lamzin, V. S. & Wilson, K. S. (1997). *Methods Enzymol.* **277**, 269–305.

Levitt, D. G. (2001). *Acta Cryst.* D**57**, 1013–1019.

MacKay, D. J. C. (2003). *Information Theory, Inference and Learning Algorithms.* Cambridge University Press.

McRee, D. E. (1992). *J. Mol. Graph.* **10**, 44–46.

Morris, R. J. (2000). PhD thesis. Karl-Franzens Universität Graz, Austria.

Morris, R. J., Perrakis, A. & Lamzin, V. S. (2002). *Acta Cryst.* D**58**, 968–975.

Morris, R. J., Perrakis, A. & Lamzin, V. S. (2004). *Methods Enzymol.* **374**, 229–244.

Morris, R. J., Zwart, P., Cohen, S., Fernandez, F. J., Kakaris, M., Kirillova, O., Vonrhein, C., Perrakis, A. & Lamzin, V. S. (2004). *J. Synchrotron Rad.* **11**, 56–59.

Oldfield, T. J. (1996). In *Crystallographic Computing 7: Proceedings of the IUCr Macromolecular Computing School, 1996*, edited by P. E. Bourne & K. D. Waterpaugh. Oxford University Press.

Oldfield, T. J. (2001*a*). *Acta Cryst.* D**57**, 82–94.

Oldfield, T. J. (2001*b*). *Acta Cryst.* D**57**, 1421–1427.

Oldfield, T. J. (2002*a*). *Acta Cryst.* D**58**, 487–493.

Oldfield, T. J. (2002*b*). *Acta Cryst.* D**58**, 963–967.

Oldfield, T. J. (2003). *Acta Cryst.* D**59**, 483–491.

Oldfield, T. J. & Hubbard, R. E. (1994). *Proteins*, **18**, 324–337.

Orengo, C. A., Mitchie, A. D., Jones, S., Jones, D. T., Swindells, M. B. & Thornton, J. M. (1997). *Structure*, **5**, 1093–1108.

Pavelcik, F., Zelinka, J. & Otwinowski, Z. (2002). *Acta Cryst.* D**58**, 275–283.

Perrakis, A., Morris, R. J. & Lamzin, V. S. (1999). *Nature Struct. Biol.* **6**, 458–463.

Proceedings of the CCP4 Study Weekend (2004). *Acta Cryst.* D**60**, 2115–2294.

Russell, S. & Norvig, P. (1996). *Artificial Intelligence – A Modern Approach.* Upper Saddle River, NJ, USA: Prentice-Hall International.

Schurmann, J. (1996). *Pattern Classification: A Unified View of Statistical and Neural Approaches.* New York: John Wiley & Sons.

Swanson, S. M. (1994). *Acta Cryst.* D**50**, 695–708.

Terwilliger, T. C. (1999). *Acta Cryst.* D**55**, 1863–1871.

Terwilliger, T. C. (2000). *Acta Cryst.* D**56**, 965–972.

Terwilliger, T. C. (2001). *Acta Cryst.* D**57**, 1755–1762.

Terwilliger, T. C. (2003*a*). *Acta Cryst.* D**59**, 38–44.

Terwilliger, T. C. (2003*b*). *Acta Cryst.* D**59**, 45–49.

Terwilliger, T. C. (2003*c*). *Acta Cryst.* D**59**, 1174–1182.

Terwilliger, T. C. (2003*d*). *Acta Cryst.* D**59**, 1688–1701.

Vapnik, V. N. (1998). *Statistical Learning Theory.* New York: John Wiley & Sons.

Zou, J.-Y. & Jones, T. A. (1996). *Acta Cryst.* D**52**, 833–841.